

```

//===== file = tcpClient.c =====
//= A message "client" program to demonstrate sockets programming
=
//=====
==
//= Notes:
=
//= 1) This program conditionally compiles for Winsock and BSD sockets.
=
//= Set the initial #define to WIN or BSD as appropriate.
=
//= 2) This program needs tcpServer to be running on another host.
=
//= Program tcpServer must be started first.
=
//= 3) This program assumes that the IP address of the host running
=
//= tcpServer is defined in "#define IP_ADDR"
=
//=
//= Example execution: (tcpServer and tcpClient running on host 127.0.0.1)
=
//= Received from server: This is a message from SERVER to CLIENT
=
//-----
--
//= Build: bcc32 tcpClient.c or cl tcpClient.c wsock32.lib for Winsock
=
//= gcc tcpClient.c -lnsl for BSD
=
//-----
--
//= Execute: tcpClient
=
//-----
--
//=====
==

#define WIN // WIN for Winsock and BSD for BSD sockets

//----- Include files -----
--
#include <stdio.h> // Needed for printf()
#include <string.h> // Needed for memcpy() and strcpy()
#include <stdlib.h> // Needed for exit()
#ifdef WIN
#include <windows.h> // Needed for all Winsock
#endif
#ifdef BSD
#include <sys/types.h> // Needed for sockets
#include <netinet/in.h> // Needed for sockets
#include <sys/socket.h> // Needed for sockets
#include <arpa/inet.h> // Needed for sockets
#include <fcntl.h> // Needed for sockets
#include <netdb.h> // Needed for sockets
#endif

```

```

//----- Defines -----
--
#define PORT_NUM      1050    // Port number used at the server
#define IP_ADDR      "127.0.0.1" // IP address of server (** HARDWIRED **)

//===== Main program
=====
int main()
{
#ifdef WIN
    WORD wVersionRequested = MAKEWORD(1,1);    // for WSA functions
    WSADATA wsaData;                          // for WSA functions
#endif
    int client_s;                               // Client socket descriptor
    struct sockaddr_in server_addr;             // Server Internet address
    char out_buf[4096];                         // Output buffer for data
    char in_buf[4096];                         // Input buffer for data
    int retcode;                                // Return code

#ifdef WIN
    // This initializes winsock
    WSASStartup(wVersionRequested, &wsaData);
#endif

    // >>> Step #1 <<<
    // Create a client socket
    // - AF_INET is Address Family Internet and SOCK_STREAM is streams
    client_s = socket(AF_INET, SOCK_STREAM, 0);
    if (client_s < 0)
    {
        printf("*** ERROR - socket() failed \n");
        exit(-1);
    }

    // >>> Step #2 <<<
    // Fill-in the server's address information and do a connect with the
    // listening server using the client socket - the connect() will block.
    server_addr.sin_family = AF_INET;           // Address family to use
    server_addr.sin_port = htons(PORT_NUM);     // Port num to use
    server_addr.sin_addr.s_addr = inet_addr(IP_ADDR); // IP address to use
    retcode = connect(client_s, (struct sockaddr *)&server_addr,
        sizeof(server_addr));
    if (retcode < 0)
    {
        printf("*** ERROR - connect() failed \n");
        exit(-1);
    }

    // >>> Step #3 <<<
    // Receive from the server using the client socket
    retcode = recv(client_s, in_buf, sizeof(in_buf), 0);
    if (retcode < 0)
    {
        printf("*** ERROR - recv() failed \n");
        exit(-1);
    }
}

```

```

// Output the received message
printf("Received from server: %s \n", in_buf);

// >>> Step #4 <<<
// Send to the server using the client socket
strcpy(out_buf, "This is a reply message from CLIENT to SERVER");
retcode = send(client_s, out_buf, (strlen(out_buf) + 1), 0);
if (retcode < 0)
{
    printf("*** ERROR - send() failed \n");
    exit(-1);
}

// >>> Step #5 <<<
// Close the client socket
#ifdef WIN
retcode = closesocket(client_s);
if (retcode < 0)
{
    printf("*** ERROR - closesocket() failed \n");
    exit(-1);
}
#endif
#ifdef BSD
retcode = close(client_s);
if (retcode < 0)
{
    printf("*** ERROR - close() failed \n");
    exit(-1);
}
#endif

#ifdef WIN
// Clean-up winsock
WSACleanup();
#endif

// Return zero and terminate
return(0);
}

```